



ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

Laboratorio di Sicurezza Informatica

Sniffing, Arp e Dhcp Spoofing, DoS

Marco Prandini

Andrea Melis

Dipartimento di Informatica – Scienza e Ingegneria

Agenda

■ Sniffing e Spoofing

- Setting laboratorio network con docker
- Leggere il traffico
- Wireshark

■ Attacchi

- Arp Spoofing
- Arp Cache Poisoning
- Dhcp Spoofing
- Denial of Service



Composizione Networking Lab

- Per questo lab utilizzeremo una serie di container, collegati tra loro attraverso una docker network.
- Utilizzeremo quindi dei container e li lanceremo attraverso il tool docker-compose



Docker Compose

- Docker Compose è uno strumento sviluppato per aiutare a definire e condividere applicazioni multi-container. Con Compose, possiamo creare un file YAML per definire i servizi e con un solo comando, possiamo far girare tutto o smontare tutto.
- In un unico file di configurazione, `docker-compose.yml` è quindi contenuto tutto il setup.
- Si definiscono Host, Reti, connessioni tra gli host.



Docker Compose

■ Con “services” si definiscono i servizi da eseguire con parametri come:

- image: l'immagine del container
- container_name:
- tty:
- cap_add:
- networks:
- Command:

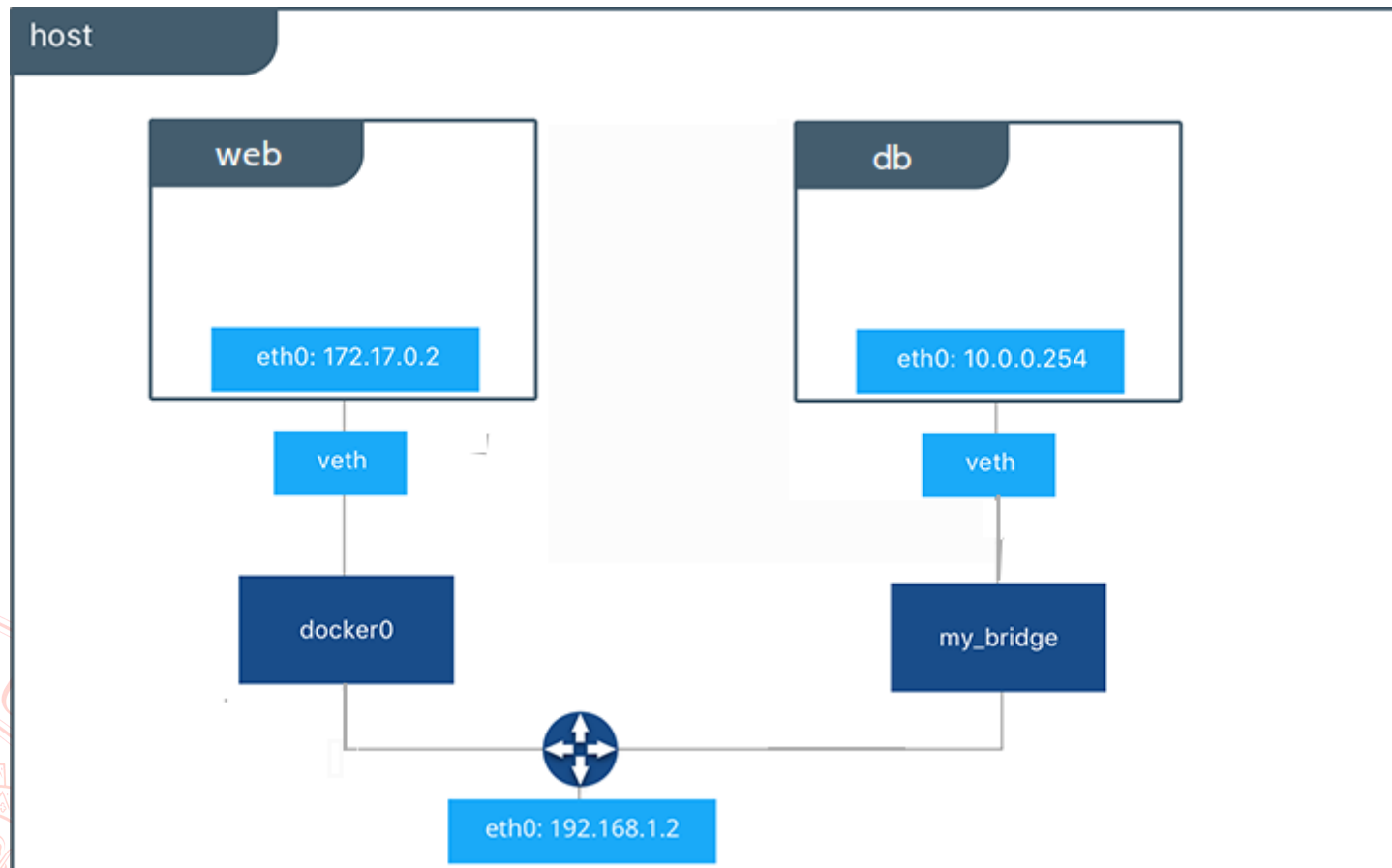
■ Con “networks” una rete:

- networks:
- nome_rete:
- name:
- ipam:
- config:
- subnet:



Rete Docker

- Rete di tipo “self named bridge”, da non confondere con il bridge di default di docker.

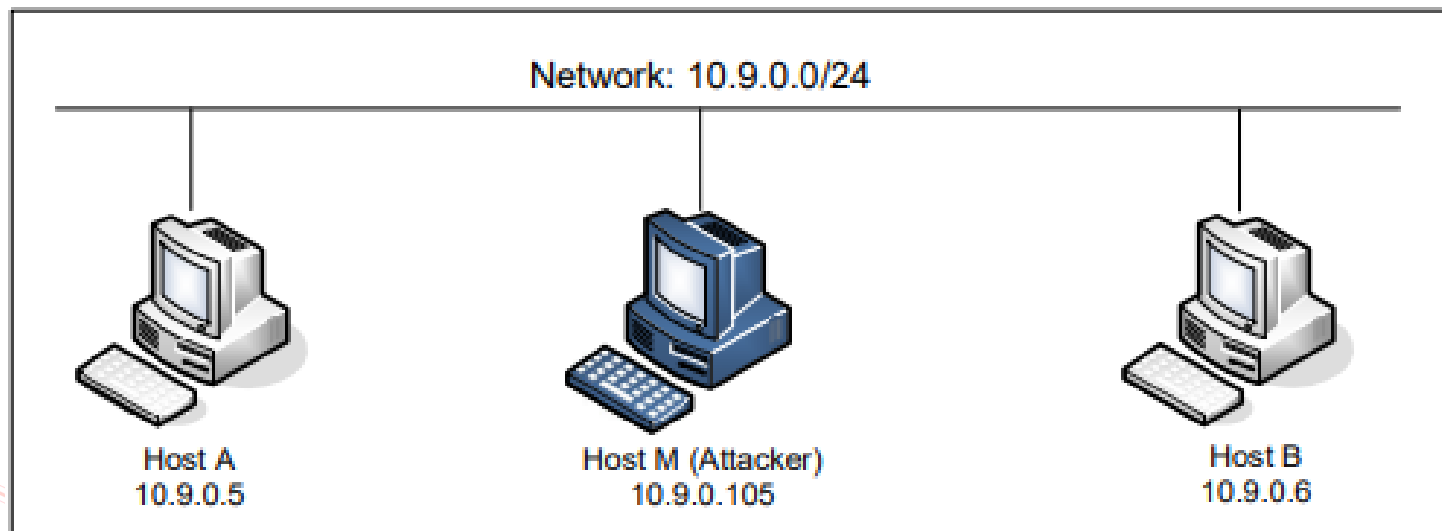


Comandi utili docker / docker-compose

- Sulla macchina del lab usarli da utente privilegiato (shell di root o sudo)
- **\$ sudo docker COMANDO**
 - ps -a # lista container e loro stato
 - kill CONTAINER # termina il container
 - rm CONTAINER # elimina container
 - exec -it CONTAINER COMANDO # esegue comando su cont.
- **\$ sudo docker-compose COMANDO**
 - build # lista container e loro stato
 - up # lancia l'infrastruttura
 - down # termina l'infrastruttura
 - rm # rimuove container spenti

Rete del Lab

- Due host A e B (Alice and Bob) e un terzo host che funge da MITM, tutti collegati alla stessa rete self-name bridged.



Creiamo il lab.

- Usiamo i comandi di docker

sudo docker-compose up

- A questo punto per completare il lab lanciamo lo script lasciato in consegna setup_infra.sh, dandogli la password di kali quando chiede permessi privilegiati con sudo

./setup_infra.sh

N.B. Potrebbe essere necessario dare i permessi di esecuzione allo script col comando

chmod +x setup_infra.sh



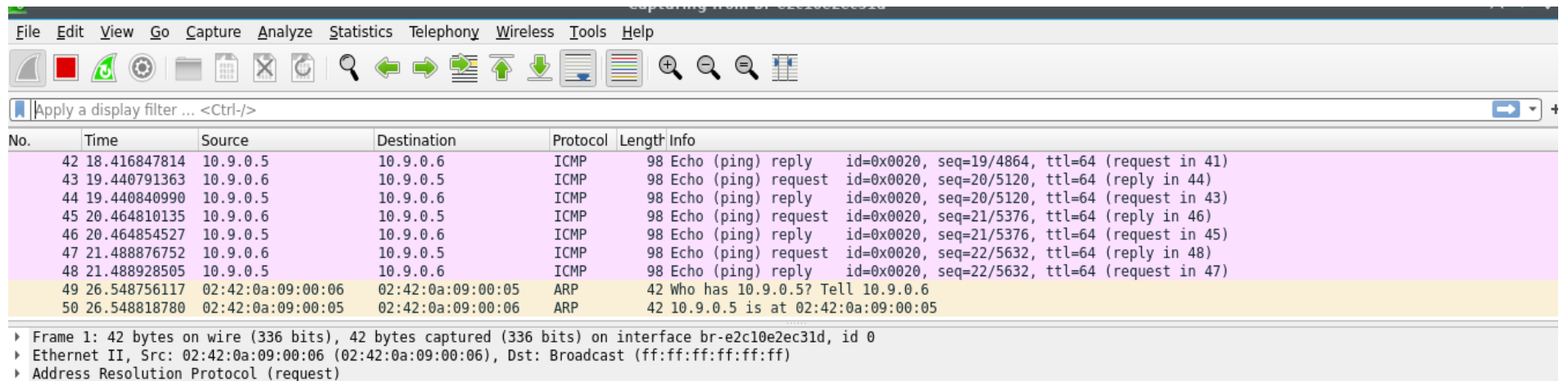
Utilizzo

- Da questo momento in poi, a parte durante l'utilizzo di wireshark utilizzeremo sempre direttamente i container.
- Connettiamoci ad un container con
hostA
hostB
hostM
- Avremo quindi una shell su ogni container.



Wireshark

■ Tipica Interfaccia Wireshark

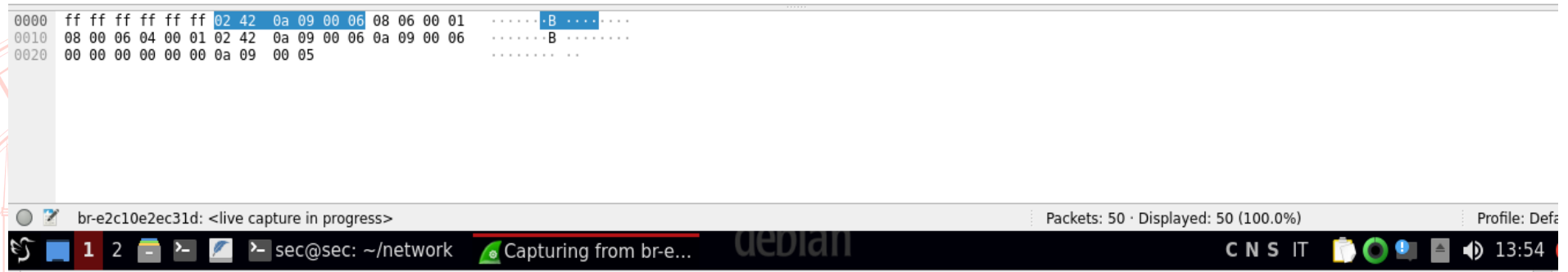


File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

Apply a display filter ... <Ctrl-/>

No.	Time	Source	Destination	Protocol	Length	Info
42	18.416847814	10.9.0.5	10.9.0.6	ICMP	98	Echo (ping) reply id=0x0020, seq=19/4864, ttl=64 (request in 41)
43	19.440791363	10.9.0.6	10.9.0.5	ICMP	98	Echo (ping) request id=0x0020, seq=20/5120, ttl=64 (reply in 44)
44	19.440840990	10.9.0.5	10.9.0.6	ICMP	98	Echo (ping) reply id=0x0020, seq=20/5120, ttl=64 (request in 43)
45	20.464810135	10.9.0.6	10.9.0.5	ICMP	98	Echo (ping) request id=0x0020, seq=21/5376, ttl=64 (reply in 46)
46	20.464854527	10.9.0.5	10.9.0.6	ICMP	98	Echo (ping) reply id=0x0020, seq=21/5376, ttl=64 (request in 45)
47	21.488876752	10.9.0.6	10.9.0.5	ICMP	98	Echo (ping) request id=0x0020, seq=22/5632, ttl=64 (reply in 48)
48	21.488928505	10.9.0.5	10.9.0.6	ICMP	98	Echo (ping) reply id=0x0020, seq=22/5632, ttl=64 (request in 47)
49	26.548756117	02:42:0a:09:00:06	02:42:0a:09:00:05	ARP	42	Who has 10.9.0.5? Tell 10.9.0.6
50	26.548818780	02:42:0a:09:00:05	02:42:0a:09:00:06	ARP	42	10.9.0.5 is at 02:42:0a:09:00:05

▶ Frame 1: 42 bytes on wire (336 bits), 42 bytes captured (336 bits) on interface br-e2c10e2ec31d, id 0
▶ Ethernet II, Src: 02:42:0a:09:00:06 (02:42:0a:09:00:06), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
▶ Address Resolution Protocol (request)



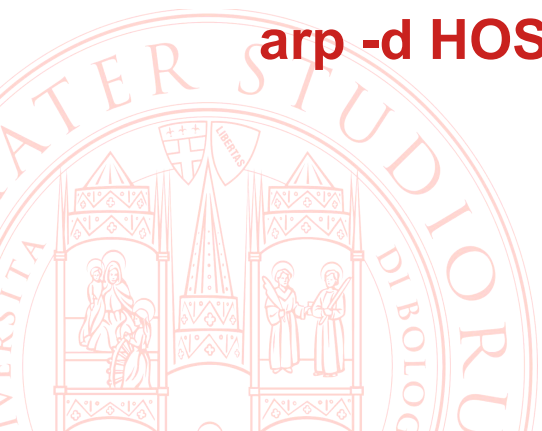
0000 ff ff ff ff ff ff 02 42 0a 09 00 06 08 06 00 01B.....
0010 08 00 06 04 00 01 02 42 0a 09 00 06 0a 09 00 06B.....
0020 00 00 00 00 00 00 0a 09 00 05
.....

br-e2c10e2ec31d: <live capture in progress> Packets: 50 · Displayed: 50 (100.0%) Profile: Defa

sec@sec: ~/network Capturing from br-e... C N S IT 13:54

Arp Spoofing

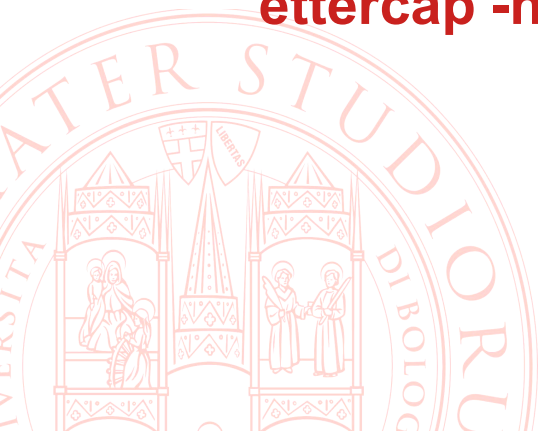
- Principale tecnica nell'ambito delle switched lan per effettuare un attacco di Man-in-the-Middle.
- Scopo dell'attacco è fare un MITM tra Alice (A) e Bob (B)
- Guardiamo le tabelle di arp sui due host con:
arp -n
- Se necessario rimuovere una entry con
arp -d HOST



Arp Poisoning (DoS ?)

- Facciamo prima il poisoning di TUTTI gli host della rete.
- Questo comporta reindirizzare TUTTO il traffico di tutti gli host della rete verso l'attaccante.
- Da fare sono in ambiente di test come il nostro
- Per gli attacchi del nostro lab utilizzeremo la suite ettercap che abbiamo installato precedentemente.

ettercap -help



Arp Poisoning (DoS ?)

- Lanciamo l'attacco con:

ettercap -T -M arp /// ///

- Verifichiamo quindi:

- Tabelle arp sugli host!
- Traffico su Wireshark
- Testiamo la connessione tra A e B!



Arp Poisoning Mirato

- Lanciamo l'attacco con:

ettercap -T -M arp /10.9.0.6// /10.9.0.5//

- Verifichiamo quindi di essere riusciti a modificare solo le tabelle arp che ci interessano per realizzare un MITM!

- A questo punto verifichiamo:

- Tabelle arp sugli host!
- Traffico su Wireshark
- Testiamo la connessione tra A e B!



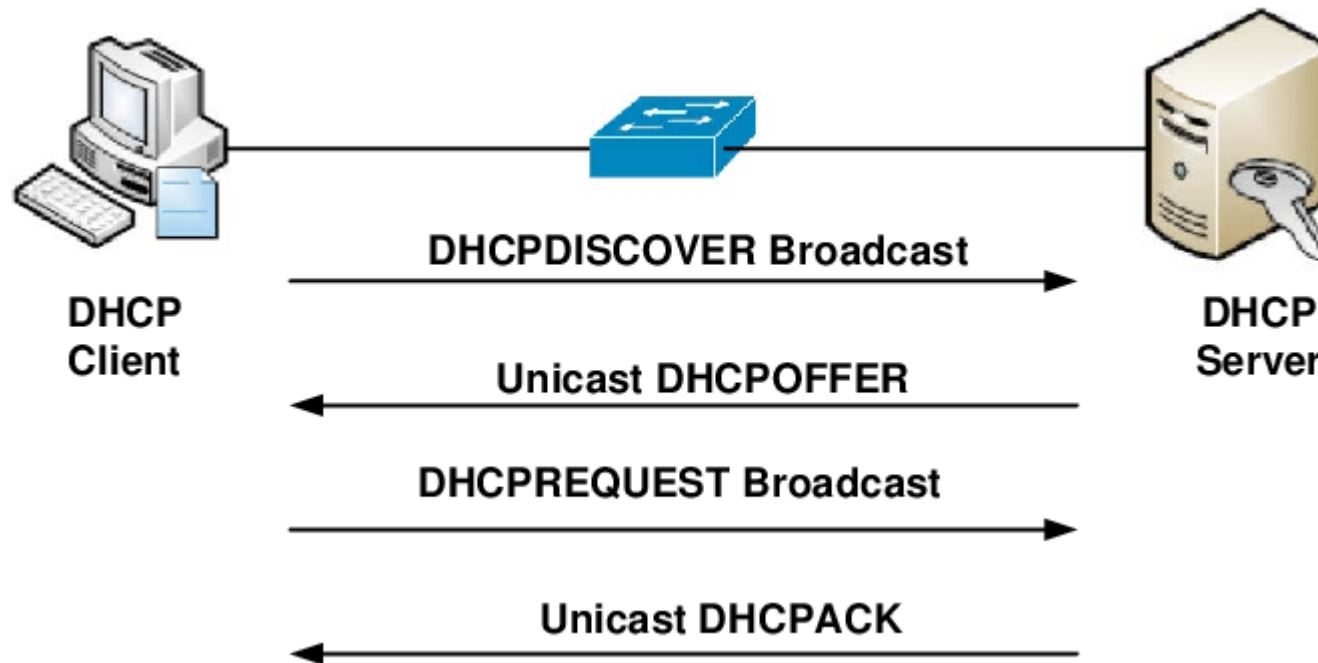
Dhcp spoofing

- Dhcp spoofing consiste nel “battere” il dhcp di default e assegnare alla vittima un ip con il gateway modificato, in modo tale da poter diventare il gateway della vittima
- “Battere” perché l’obiettivo è lanciare tante risposte dhcp, fino a che non si stabilisca una connessione.
- Seguendo lo stesso approccio è possibile anche effettuare un DoS, inondando di request la rete, in modo tale che il server dhcp venga saturato.
- Lanciamo l’attacco con:
ettercap -T -M dhcp:10.9.0.20-60/255.255.255.0/8.8.8.8



Dhcp spoofing

- Principio funzionamento dhcp è il seguente.



Dhcp spoofing, preparazione ed esecuzione

- Rimuoviamo l'ip al docker B
ip addr del 10.9.0.6/24 dev eth0
- Sulla macchina .105 (l'attacker) lanciare
ettercap -T -M dhcp:10.9.0.20-60/255.255.255.0/8.8.8.8
- Sulla macchina B lanciamo
dhclient eth0
- Come al solito, guardiamo wireshark e cerchiamo di capire cosa è successo
 - Che pacchetti “girano”?
 - Quale indirizzo viene assegnato all'host B ?

DoS

- Come ultimo esercizio proviamo a simulare un attacco di Denial of Service di tipo SYN flood
- Il SYN flood è un attacco di tipo denial of service nel quale un utente malevolo invia una serie di richieste SYN-TCP verso il sistema oggetto dell'attacco.
- Scopo dell'attaccante è aprire un gran numero di connessioni tcp sulla vittima in modo tale da saturarne la banda
- È possibile utilizzare in questo caso un ip “spoofed” in modo tale da creare una connessione per ogni ip.



DoS

- Per eseguire quest'attacco utilizzeremo un tool, hping3 già installato sul docker dell'attaccante
- Allo stesso tempo monitoreremo lo stato della connessione tra Alice e Bob con un altro tool, che misura il bandwidth disponibile in una connessione, iperf.
- Testiamo prima il bandwidth senza nessun attacco
- Sul container Bob lanciare
`iperf -s`
- Sul container Alice lanciare
`iperf -c 10.9.0.6`

DoS

- A questo punto utilizziamo hping3 dalla macchina attaccante per lanciare il SYN flood attack

- Lanciamo l'attacco con:

```
hping3 -c 10000 -d 120 -S -w 64 -p 21 --flood --rand-source 10.9.0.6
```

- Dove:

-c 100000 = Number of packets to send.

-d 120 = Size of each packet that was sent to target machine.

-S = I am sending SYN packets only.

-w 64 = TCP window size.

-p = Destination port

--flood = Sending packets as fast as possible, without taking care to show incoming replies. Flood mode.

--rand-source = Using Random Source IP Addresses. You can also use -a or --spoof to hide hostnames. See MAN page below.

DoS

- **Monitoriamo la connessione con iperf e vediamo se le performance degradano!**
- **hping3 supporta anche altri metodi di attacco, ad esempio.**
 - **LAND attack**
 - **SMURF attack**
 - **..**

